4.14 Non-exam assessment – the computing practical project

4.14.1 Overview

4.14.1.1 Purpose of the project

The project allows students to develop their practical skills in the context of solving a realistic problem or carrying out an investigation. The project is intended to be as much a learning experience as a method of assessment; students have the opportunity to work independently on a problem of interest over an extended period, during which they can extend their programming skills and deepen their understanding of computer science.

The most important skill that should be assessed through the project is a student's ability to create a programmed solution to a problem or investigation. This is recognised by allocating 42 of the 75 available marks to the technical solution and a lower proportion of marks for supporting documentation to reflect the expectation that reporting of the problem, its analysis, the design of a solution or plan of an investigation and testing and evaluation will be concise.

4.14.1.2 Types of problem/investigation

Students are encouraged to choose a problem to solve or investigate that will interest them and that relates to a field that they have some knowledge of. There are no restrictions on the types of problem/investigation that can be submitted or the development tools (for example programming language) that can be used. The two key questions to ask when selecting a problem/investigation are:

- Does the student have existing knowledge of the field, or are they in a position to find out about it?
- Is a solution to the problem/investigation likely to give the student the opportunity to demonstrate the necessary degree of technical skill to achieve a mark that reflects their potential?

Some examples of the types of problem to solve or investigate are:

- a simulation for example, of a business or scientific nature, or an investigation of a well-known problem such as the game of life
- a solution to a data processing problem for an organisation, such as membership systems
- the solution of an optimisation problem, such as production of a rota, shortest-path problems or route finding
- a computer game
- an application of artificial intelligence
- a control system, operated using a device such as an Arduino board
- a website with dynamic content, driven by a database back-end
- an app for a mobile phone or tablet
- an investigation into an area of computing, such as rendering a three-dimensional world on screen
- investigating an area of data science using, for example, Twitter feed data or online public data sets
- investigating machine learning algorithms.

There is an expectation that within a centre, the problems chosen by students to solve or investigate will be sufficiently different to avoid the work of one student informing the work of another because they are working on the same problem or investigation. Teachers will be required to record on the Candidate record form for each student that they have followed this guideline. If in any doubt on whether problems chosen by students have the potential to raise this issue, please contact your AQA advisor.

Table 1 and Table 2 show the technical skills and coding styles required for an A-level standard project. If a problem/investigation is selected that is not of A-level standard then the marks available in each section will be restricted.

4.14.1.3 Project documentation structure

The project is assessed in five sections. The table below lists the maximum available mark for each section of the project:

Section		Max mark
1	Analysis	9
2	Documented design	12
3	Technical solution	42
4	Testing	8
5	Evaluation	4
Total		**75**

For marking purposes, the project documentation should be presented in the order indicated in the table above. The table does not imply that students are expected to follow a traditional systems life cycle approach when working on their projects, whereby a preceding stage must be completed before the next can be tackled. It is recognised that this approach is unsuited to the vast majority of project work, and that project development is likely to be an iterative process, with earlier parts of the project being revisited as a result of discoveries made in later parts. Students should be encouraged to start prototyping and writing code early on in the project process. A recommended strategy is to tackle the critical path early in the project development process. The critical path is the part of the project that everything else depends on for a working system or a complete investigation result to be achieved.

4.14.2 Using a level of response mark scheme

Level of response mark schemes are broken down into a number of levels, each of which has a descriptor. The descriptor for the level shows the average performance for the level. There are a range of marks in each level. The descriptor for the level represents a typical mid-mark performance in that level.

Before applying the mark scheme to a student's project, read it through and annotate it to show the qualities that are being looked for. You can then apply the mark scheme.

4.14.2.1 Step 1. Determine a level

Start at the lowest level of the mark scheme and use it as a ladder to see whether the performance in that section of the project meets the descriptor for that level. The descriptor for the level indicates the different qualities that might be seen in the student's work for that level. If it meets the lowest level then go to the next one and decide if it meets this level, and so on, until you have a match between the level descriptor and the work. With practice and familiarity you will find you will be able to quickly skip through the lower levels of the mark scheme.

When assigning a level you should look at the overall quality of the work rather than any small or specific parts where the student has not performed quite as the level descriptor. If the work covers different aspects of different levels of the mark scheme you should use a best fit approach for defining the level and then use the variability of the response to help decide the mark within the level. ie if the response is predominantly level 3 with a small amount of level 4 material it would be placed in level 3 but be awarded a mark near the top of the level because of the level 4 content.

4.14.2.2 Step 2. Determine a mark

Once you have assigned a level you need to decide on the mark. The exemplar materials used for standardisation will help. This work will have been awarded a mark by AQA. You can compare your student's work with the exemplar to determine if it is the same standard, better or worse. You can then use this to allocate a mark for the work based on AQA's mark on the exemplar.

You may well need to read back through the work as you apply the mark scheme to clarify points and assure yourself that the level and the mark are appropriate.

Work which contains nothing of relevance to the project area being assessed must be awarded no marks for that area.

4.14.3 Marking criteria

4.14.3.1 Analysis (9 marks)

Level	Mark range	Description
3	7–9	Fully or nearly fully scoped analysis of a real problem, presented in a way that a third party can understand. Requirements fully documented in a set of measurable and appropriate specific objectives, covering all required functionality of the solution or areas of investigation. Requirements arrived at by considering, through dialogue, the needs of the intended users of the system, or recipients of the outcomes for investigative projects. Problem sufficiently well modelled to be of use in subsequent stages.
2	4–6	Well scoped analysis (but with some omissions that are not serious enough to undermine later design) of a real problem. Most, but not all, requirements documented in a set of, in the main, measurable and appropriate specific objectives that cover most of the required functionality of a solution or areas of investigation. Requirements arrived at, in the main, by considering, through dialogue, the needs of the intended users of the system, or recipients of the outcomes for investigative projects. Problem sufficiently well modelled to be of use in subsequent stages.
1	1–3	Partly scoped analysis of a problem. Requirements partly documented in a set of specific objectives, not all of which are measurable or appropriate for developing a solution. The required functionality or areas of investigation are only partly addressed. Some attempt to consider, through dialogue, the needs of the intended users of the system, or recipients of the outcomes for investigative projects. Problem partly modelled and of some use in subsequent stages.

4.14.3.2 Documented design (12 marks)

Level	Mark range	Description
4	10–12	Fully or nearly fully articulated design for a real problem, that describes how all or almost all of the key aspects of the solution/investigation are to be structured/are structured.
3	7–9	Adequately articulated design for a real problem that describes how most of the key aspects of the solution/investigation are to be structured/are structured.
2	4–6	Partially articulated design for a real problem that describes how some aspects of the solution/investigation are to be structured/are structured.
1	1–3	Inadequate articulation of the design of the solution so that it is difficult to obtain a picture of how the solution/investigation is to be structured/ is structured without resorting to looking directly at the programmed solution.

4.14.3.3 Technical solution (42 marks)

4.14.3.3.1 Completeness of solution (15 marks)

Level	Mark range	Description
3	11–15	A system that meets almost all of the requirements of a solution/an investigation (ignoring any requirements that go beyond the demands of A-level).
2	6–10	A system that achieves many of the requirements but not all. The marks at the top end of the band are for systems that include some of the most important requirements.
1	1–5	A system that tackles some aspects of the problem or investigation.

4.14.3.3.2 Techniques used (27 marks)

Level	Mark range	Description	Additional information
3	19–27	The techniques used are appropriate and demonstrate a level of technical skill equivalent to those listed in Group A in Table 1. Program(s) demonstrate(s) that the skill required for this level has been applied sufficiently to demonstrate proficiency.	Above average performance: Group A equivalent algorithms and model programmed more than well to excellent; all or almost all excellent coding style characteristics; more than to highly effective solution. Average performance: Group A equivalent algorithms and/or model programmed well; majority of excellent coding style characteristics; an effective solution. Below average performance: Group A equivalent algorithms and/or model programmed just adequately to fully adequate; some excellent coding style characteristics; less than effective to fairly effective solution.
2	10–18	The techniques used are appropriate and demonstrate a level of technical skill equivalent to those listed in Group B in Table 1. Program(s) demonstrate(s) that the skill required for this level has been applied sufficiently to demonstrate proficiency.	Above average performance: Group B equivalent algorithms and model programmed more than well to excellent; majority of excellent coding style characteristics; more than to highly effective solution. Average performance: Group B equivalent algorithms and/or model programmed well; some excellent coding style characteristics; effective solution. Below average performance: Group B equivalent algorithms and/or model programmed just adequately to fully adequate; all or almost all relevant good coding style characteristics but possibly one example at most of excellent characteristics; less than effective to fairly effective solution.
1	1–9	The techniques used demonstrate a level of technical skill equivalent to those listed in Group C in Table 1. Program(s) demonstrate(s) that the skill required for this level has been applied sufficiently to demonstrate proficiency.	Above average performance: Group C equivalent model and algorithms programmed more than well to excellent; almost all relevant good coding style characteristics; more than to highly effective simple solution. Average performance: Group C equivalent model and algorithms programmed well; some relevant good coding style characteristics; effective simple solution. Below average performance: Group C equivalent algorithms and/or model programmed in a severely limited to limited way; basic coding style characteristics; trivial to lacking in effectiveness simple solution.

Select the band, 1, 2 or 3 with level of demand description that best matches the techniques and skill that the student's program attempts to cover. The emphasis is on what the student has actually achieved that demonstrates proficiency at this level rather than what the student has set out to use and do but failed to demonstrate, eg because of poor execution. Check the proficiency demonstrated in the program. If the student fails to demonstrate proficiency at the initial level of choice, drop down a level to see if what the student has done demonstrates proficiency at this level for the lower demand until a match is obtained. Table 1 is indicative of the standard required and is not to be treated as just a list of things for students to select from and to be automatically credited for including in their work.

As indicated above, having selected the appropriate level for techniques used and proficiency in their use, the exact mark to award should be determined based upon:

- the extent to which the criteria for the mark band have been achieved
- the quality of the coding style that the student has demonstrated (see Table 2 for exemplification of what is expected)
- the effectiveness of the solution.

4.14.3.4 Example technical skills

4.14.3.4.1 Table 1: Example technical skills

Group	Model (including data model/structure)	Algorithms
A	Complex data model in database (eg several interlinked tables)	Cross-table parameterised SQL
		Aggregate SQL functions
		User/CASE-generated DDL script
	Hash tables, lists, stacks, queues, graphs, trees or structures of equivalent standard	Graph/Tree Traversal
	Files(s) organised for direct access	List operations
		Linked list maintenance
		Stack/Queue Operations
		Hashing
	Complex scientific/mathematical/robotics/ control/business model	Advanced matrix operations
		Recursive algorithms
		Complex user-defined algorithms (eg optimisation, minimisation, scheduling, pattern matching) or equivalent difficulty
		Mergesort or similarly efficient sort
	Complex user-defined use of object-orientated programming (OOP) model, eg classes, inheritance, composition, polymorphism, interfaces	Dynamic generation of objects based on complex user-defined use of OOP model
	Complex client-server model	Server-side scripting using request and response objects and server-side extensions for a complex client-server model
		Calling parameterised Web service APIs and parsing JSON/XML to service a complex client-server model

Group	Model (including data model/structure)	Algorithms
B	Simple data model in database (eg two or three interlinked tables) Multi-dimensional arrays Dictionaries Records Text files File(s) organised for sequential access Simple scientific/mathematical /robotics/ control/business model Simple OOP model Simple client-server model	Single table or non-parameterised SQL Bubble sort Binary search Writing and reading from files Simple user defined algorithms (eg a range of mathematical/statistical calculations) Generation of objects based on simple OOP model Server-side scripting using request and response objects and server-side extensions for a simple client-server model Calling Web service APIs and parsing JSON/ XML to service a simple client-server model
C	Single-dimensional arrays Appropriate choice of simple data types Single table database	Linear search Simple mathematical calculations (eg average) Non-SQL table access

Note that the contents of Table 1 are examples, selected to illustrate the level of demand of the technical skills that would be expected to be demonstrated in each group. The use of alternative algorithms and data models is encouraged. If a project cannot easily be marked against Table 1 (for example, a project with a considerable hardware component) then please consult your AQA non-exam assessment adviser or provide a full explanation of how you have arrived at the mark for this section when submitting work for moderation.

4.14.3.4.2 Table 2: Coding styles

Style	Characteristic
Excellent	Modules (subroutines) with appropriate interfaces Loosely coupled modules (subroutines) – module code interacts with other parts of the program through its interface only Cohesive modules (subroutines) – module code does just one thing Modules(collections of subroutines) – subroutines with common purpose grouped Defensive programming Good exception handling
Good	Well-designed user interface Modularisation of code Good use of local variables Minimal use of global variables Managed casting of types Use of constants Appropriate indentation Self-documenting code Consistent style throughout File paths parameterised
Basic	Meaningful identifier names Annotation used effectively where required

The descriptions in Table 2 are cumulative, ie for a program to be classified as excellent it would be expected to exhibit the characteristics listed as excellent, good and basic not just those listed as excellent.

4.14.3.5 Testing (8 marks)

Level	Mark range	Description
4	7–8	Clear evidence, in the form of carefully selected representative samples, that thorough testing has been carried out. This demonstrates the robustness of the complete or nearly complete solution/thoroughness of investigation and that the requirements of the solution/investigation have been achieved.
3	5–6	Extensive testing has been carried out, but the evidence presented in the form of representative samples does not make clear that all of the core requirements of the solution/investigation have been achieved. This may be due to some key aspects not being tested or because the evidence is not always presented clearly.
2	3–4	Evidence in the form of representative samples of moderately extensive testing, but falling short of demonstrating that the requirements of the solution/investigation have been achieved and the solution is robust/investigation thorough. The evidence presented is explained.
1	1–2	A small number of tests have been carried out, which demonstrate that some parts of the solution work/some outcomes of the investigation are achieved. The evidence presented may not be entirely clear.

Evidence for the testing section may be produced after the system has been fully coded or during the coding process. It is expected that tests will either be planned in a test plan or that the tests will be fully explained alongside the evidence for them. Only carefully selected representative samples are required.

4.14.3.6 Evaluation (4 marks)

Level	Mark	Description
4	4	Full consideration given to how well the outcome meets all of its requirements. How the outcome could be improved if the problem was revisited is discussed and given detailed consideration. Independent feedback obtained of a useful and realistic nature, evaluated and discussed in a meaningful way.
3	3	Full or nearly full consideration given to how well the outcome meets all of its requirements. How the outcome could be improved if the problem was revisited is discussed but consideration given is limited. Independent feedback obtained of a useful and realistic nature but is not evaluated and discussed in a meaningful way, if at all.
2	2	The outcome is discussed but not all aspects are fully addressed either by omission or because some of the requirements have not been met and those requirements not met have been ignored in the evaluation. No independent feedback obtained or if obtained is not sufficiently useful or realistic to be evaluated in a meaningfully way even if attempted.
1	1	Some of the outcomes are assessed but only in a superficial way. No independent feedback obtained or if obtained is so basic as to be not worthy of evaluation.

4.14.4 Project tasks that are not of A-level standard

If the task (problem or investigation) selected for a project is not of A-level standard, mark the project against the criteria given, but adjust, the mark awarded downwards by two marking levels (two marks in the case of evaluation) in each section for all but the technical solution. You should have already taken the standard into account for this, by directly applying the criteria. For example, if a student had produced a 'fully or nearly fully articulated design of a real problem describing how solution is to be structured/is structured'. This would, for an A-level standard project, achieve a mark in level 4 for Documented Design (10 – 12 marks). If the problem selected was too simple to be of A-level standard but the same criteria had been fulfilled, shift the mark awarded down by two levels, into level 2, an award of 4 – 6 marks. If a downward shift by two levels is not possible, then a mark in the lowest level should be awarded.

4.14.5 Guide to non-exam assessment documentation

4.14.5.1 Analysis

Students are expected to:

- produce a clear statement that describes the problem area and specific problem that is being solved/investigated
- outline how they researched the problem
- state for whom the problem is being solved/investigated
- provide background in sufficient detail for a third party to understand the problem being solved/investigated
- produce a numbered list of measurable, "appropriate" specific objectives, covering all required functionality of the solution or areas of investigation (Appropriate means that the specific objectives are single purpose and at a level of detail that is without ambiguity)
- report any modelling of the problem that will inform the Design stage, for example a graph/network model of Facebook connections or an E-R model.

A fully scoped analysis is one that has:

- researched the problem thoroughly
- has clearly defined the problem being solved/investigated
- omitted nothing that is relevant to subsequent stages
- statements of objectives which clearly and unambiguously identify the scope of the project
- modelled the problem for the Design stage where this is possible and necessary.

4.14.5.2 Design

Students are expected to articulate their design in a manner appropriate to the task and with sufficient clarity for a third party to understand how the key aspects of the solution/investigation are structured and on what the design will rely, eg use of numerical and scientific package libraries, data visualisation package library, particular relational database and/or web design framework. The emphasis is on communicating the design; therefore it is acceptable to provide a description of the design in a combination of diagrams and prose as appropriate, as well as a description of algorithms, SQL, data structures, database relations as appropriate, and using relevant technical description languages, such as pseudo-code. Where design of a user interface is relevant, screen shots of actual screens are acceptable.

4.14.5.3 Technical solution

Students should provide program listing(s) that demostrate their technical skill. The program listing(s) should be appropriately annotated and self-documenting (an approach that uses meaningful identifiers, with well structured code that minimises instances where program comments are necessary).

Students should present their work in a way that will enable a third party to discern the quality and purpose of the coding. This could take the form of:

- an overview guide which amongst other things includes the names of entities such as executables, data filenames/urls, database names, pathnames so that a third party can, if they so desire, run the solution/investigation
- explanations of particularly difficult-to-understand code sections; a careful division of the presentation of the code listing into appropriately labelled sections to make navigation as easy as possible for a third party reading the code listing.

Achievement of the latter, to an extent, is linked to the skill in applying a structured approach during the course of developing the solution or carrying out the investigation.

4.14.5.4 Testing

Students must provide and present in a structured way for example in tabular form, clear evidence of testing. This should take the form of carefully selected and representative samples, which demonstrate the robustness of the complete, or nearly complete, solution/thoroughness of investigation and which demonstrate that the requirements of the solution/investigation have been achieved. The emphasis should be on producing a representative sample in a balanced way and not on recording every possible test and test outcome. Students should explain the tests carried out alongside the evidence for them. This could take the form of:

- an introduction and overview
- the test performed
- its purpose if not self-evident
- the test data
- the expected test outcome
- the actual outcome with a sample of the evidence, for example screen shots of before and after the test, etc, sampled in order to limit volume.

4.14.5.5 Evaluation

Students should consider and assess how well the outcome meets its requirements. Students should obtain independent feedback on how well the outcome meets its requirements and discuss this feedback. Some of this feedback could be generated during prototyping. If so, this feedback, and how/ why it was taken account must be presented and referenced so it can be found easily.

Students should also consider and discuss how the outcome could be improved more realistically if the problem/investigation were to be revisited.

4.14.6 Assessment objective breakdown for non-exam assessment

Section	Total	AO2	AO3	Elements
Analysis	9	9		AO2b
Design	12		12	AO3a
Technical Solution	42		42	AO3b
Testing	8		8	AO3c
Evaluation	4		4	AO3c
Totals	75	9	66	